

TITLE OF THE INVENTION

MODULATION SYSTEM

BACKGROUND OF THE INVENTIONField of the Invention

5 This invention relates to a modulation method, a modulation apparatus, a demodulation method, a demodulation apparatus, an information recording medium, an information transmission method, and an information transmission apparatus.

Description of the Related Art

10 Some modulation (encoding) procedures used for digital signals recorded on recording mediums are of a (1, 7)RLL type, where "(1, 7)RLL" means run length limiting rules such that 1 to 7 successive bits of "0" should be between bits of "1" in a modulation-resultant bit stream. The (1, 7)RLL modulation tends to
15 insufficiently suppress DC and near-DC components of a modulation-resultant bit stream. Therefore, in specified conditions, the spectrum of an information signal enters a frequency band assigned to a servo signal. In this case, the information signal interferes with servo control.

20 Japanese patent application publication number 6-195887/1994 discloses first and second modulation apparatuses. The first modulation apparatus in Japanese application 6-195887 processes an input signal which has a sequence of symbols each having one byte. The first modulation apparatus includes an
25 inverting circuit, a parallel-to-serial converting circuit, and a (1, 7)RLL modulation circuit. The inverting circuit receives the input

signal, and inverts all bits in every odd-numbered symbol. The inverting circuit keeps every even-numbered symbol unchanged. The output signal from the inverting circuit is converted into a first bit stream by the parallel-to-serial converting circuit. The (1, 7)RLL modulation circuit subjects the first bit stream to (1, 7)RLL modulation, thereby generating a modulation-resultant bit stream (a second bit stream). The inversion of every odd-numbered symbol by the inverting circuit causes the suppression of a DC component of the modulation-resultant bit stream.

10 The second modulation apparatus in Japanese application 6-195887 includes a randomizing circuit and a (1, 7)RLL modulation circuit. The randomizing circuit receives an input signal, and randomizes the input signal. The randomizing circuit outputs the randomizing-resultant signal to the (1, 7)RLL modulation circuit.

15 The (1, 7)RLL modulation circuit subjects the randomizing-resultant signal to (1, 7)RLL modulation, thereby generating a modulation-resultant bit stream. The signal processing by the randomizing circuit causes the suppression of a DC component of the modulation-resultant bit stream.

20 Japanese patent application publication number 10-340543/1998 discloses (1, 7)RLL modulation provided with DSV (digital sum variation) control for suppressing DC and low-frequency components of a modulation-resultant bit stream. According to the (1, 7)RLL modulation in Japanese application 10-340543, three

25 successive bits in every prescribed position in a (1, 7)RLL code string is replaced by six successive DSV control bits of a pattern

chosen so that the rules "(1, 7)RLL" will be observed.

Japanese patent application publication number 2000-105981 discloses (1, 8)RLL modulation provided with DSV control for suppressing DC and low-frequency components of a modulation-
5 resultant bit stream. The (1, 8)RLL modulation in Japanese application 2000-105981 includes 8-12 modulation. The 8-12 modulation refers to a table containing 12-bit output code words assigned to 8-bit input code words respectively. Input data are divided into 8-bit segments each handled as an input code word.
10 Every input code word is converted into an output code word by referring to the table. Specifically, the output code word assigned to the input code word is read out from the table. As a result, the input data are converted into a modulation-resultant bit stream formed by a sequence of output code words read out from the table.
15 The output code words in the table and the output code words read out therefrom to form the modulation-resultant bit stream are designed so that the modulation-resultant bit stream will follow the rules "(1, 8)RLL". Specifically, a succession of a preliminary current output code word and a next output code word is generated in
20 response to every two successive input code words. Conditions of the connection between the preliminary current output code word and the next output code word are checked to decide whether or not the succession follows the rules "(1, 8)RLL". When it is decided that the succession does not follow the rules "(1, 8)RLL", the
25 preliminary current output code word is replaced by another current output code word.

Japanese patent application publication number 2000-286709 discloses a modulation system which includes a formatter, an 8-15 modulator, and an NRZI converter. The formatter converts an input digital signal into a second digital signal of a predetermined format.

5 The formatter outputs the second digital signal to the 8-15 modulator. The 8-15 modulator contains a set of seven different encoding tables. The 8-15 modulator converts or encodes every 8-bit block of the output digital signal from the formatter into a 15-bit code word by referring to the set of the encoding tables. The 15-bit
10 code word forms a 15-bit block of a modulation-resultant bit stream (a modulation-resultant digital signal). The 15-bit code word is chosen to enable its NRZI conversion result to follow run length limiting rules such that a minimum run length is $3T$ and a maximum run length is $11T$ where T denotes the length or period of one bit
15 (one channel bit). The 8-15 modulator outputs the modulation-resultant bit stream (the modulation-resultant digital signal) to the NRZI converter. The NRZI converter subjects the output digital signal of the 8-15 modulator to NRZI modulation, thereby generating a digital signal of an NRZI code.

20 In the modulation system of Japanese application 2000-286709, each of the encoding tables stores 15-bit code words assigned to different states of an 8-bit input block respectively. In addition, each of the encoding tables contains state information for selecting one from the encoding tables which will be used to
25 convert a next 8-bit input block. This design is to enable the NRZI conversion result of a succession of two selected 15-bit code words

to follow the run length limiting rules. The contents of the encoding tables are optimized in view of information about the frequencies of occurrence of different states of an 8-bit input block. Furthermore, first and second specified ones of the encoding tables
5 are designed so that the NRZI modulation results of 15-bit code words in the first specified encoding table which correspond to prescribed 8-bit input blocks will be opposite in polarity ("odd-even" in the number of "1") to those of 15-bit code words in the second specified encoding table.

10 In the modulation system of Japanese application 2000-286709, two candidate 15-bit code words may be selected from the first and second specified encoding tables in response to a given 8-bit input block. DSVs (digital sum variations) are calculated for the candidate 15-bit code words, respectively. The absolute values of
15 the DSVs are compared. One of the candidate 15-bit code words which corresponds to the smaller of the absolute values of the DSVs is finally selected as a 15-bit output code word. In this way, DSV control is implemented.

Japanese patent application publication number 2000-332613
20 discloses a 4-6 modulator. The 4-6 modulator contains a set of four different encoding tables. The 4-6 modulator converts or encodes every 4-bit input code word into a 6-bit output code word by referring to the set of the encoding tables. The 6-bit output code word forms a 6-bit block of a modulation-resultant bit stream. Each
25 of the encoding tables stores 6-bit output code words assigned to 4-bit input code words respectively. In addition, the encoding tables

contain next-table selection numbers accompanying the respective 6-bit output code words therein. Each of the next-table selection numbers designates one among the encoding tables which will be used to convert a next 4-bit input code word. The output code words and the next-table selection numbers in the encoding tables are designed so that the modulation-resultant bit stream formed by a succession of selected output code words will follow (1, 7)RLL. First and second specified ones of the encoding tables are designed so that 6-bit output code words in the first specified encoding table which correspond to prescribed 4-bit input code words will be opposite in polarity ("odd-even" in the number of "1") to those of 6-bit output code words in the second specified encoding table.

In the 4-6 modulator of Japanese application 2000-332613, two candidate 6-bit output code words may be selected from the first and second specified encoding tables in response to a given 4-bit input code word. DSVs (digital sum variations) are calculated for the candidate 6-bit output code words, respectively. The absolute values of the DSVs are compared. One of the candidate 6-bit output code words which corresponds to the smaller of the absolute values of the DSVs is selected as a final 6-bit output code word. In this way, DSV control is implemented.

SUMMARY OF THE INVENTION

It is a first object of this invention to provide a modulation method which is excellent in encoding rate (encoding efficiency), suppression of a DC component, and simplicity of an encoding table.

It is a second object of this invention to provide a modulation

apparatus which is excellent in encoding rate, suppression of a DC component, and simplicity of an encoding table.

It is a third object of this invention to provide an improved demodulation method.

5 It is a fourth object of this invention to provide an improved demodulation apparatus.

It is a fifth object of this invention to provide an improved information recording medium.

10 It is a sixth object of this invention to provide an information transmission method which is excellent in encoding rate, suppression of a DC component, and simplicity of an encoding table.

15 It is a seventh object of this invention to provide an information transmission apparatus which is excellent in encoding rate, suppression of a DC component, and simplicity of an encoding table.

A first aspect of this invention provides a modulation method comprising the steps of generating a 6-bit output code word in response to every 4-bit input code word by referring to a set of encoding tables, wherein the encoding tables contain output code words assigned to input code words, and contain encoding-table designation information accompanying each output code word, wherein the encoding-table designation information designates an encoding table among the encoding tables which is used next to generate an output code word immediately following the output code word accompanied with the encoding-table designation information; and sequentially connecting the generated output code

20

25

words into a sequence of the generated output code words which follows predetermined run length limiting rules (1, k)RLL, where "k" denotes a predetermined natural number between 7 and 12.

A second aspect of this invention is based on the first aspect
5 thereof, and provides a modulation method wherein NRZI
conversion results of output code words in first specified one of the
encoding tables which are assigned to prescribed input code words
are opposite in polarity to NRZI conversion results of output code
words in second specified one of the encoding tables which are
10 assigned to the prescribed input code words, and further
comprising the steps of generating a first candidate current output
code word in response to a current input code word equal to one of
the prescribed input code words by referring to the first specified
one of the encoding tables, and generating a second candidate
15 current output code word in response to the current input code
word equal to said one of the prescribed input code words by
referring to the second specified one of the encoding tables,
wherein a succession of a specified immediately-preceding output
code word and the first candidate current output code word and
20 also a succession of the specified immediately-preceding output
code word and the second candidate current output code follow the
predetermined run length limiting rules (1, k)RLL.

A third aspect of this invention is based on the second aspect
thereof, and provides a modulation method further comprising the
25 step of selecting one from the first and second candidate current
output code words as a final current output code word.

A fourth aspect of this invention is based on the second aspect thereof, and provides a modulation method further comprising the steps of calculating a first CDS of the first candidate current output code word; updating a first DSV of the first candidate current output
5 code word and previous final output code words in response to the first CDS; calculating a second CDS of the second candidate current output code word; updating a second DSV of the second candidate current output code word and previous final output code words in response to the second CDS; determining which of an absolute value
10 of the first DSV and an absolute value of the second DSV is smaller; and selecting one from the first and second candidate current output code words which corresponds to the smaller DSV absolute value as a final current output code word.

A fifth aspect of this invention is based on the first aspect
15 thereof, and provides a modulation method further comprising the steps of predicting repetition of a minimum run length at least a predetermined number of times in the sequence of the generated output code words; and when the repetition of the minimum run length is predicted, changing an output code word causing the
20 repetition to prevent the repetition of the minimum run length from occurring in the sequence of the generated output code words.

A sixth aspect of this invention provides a modulation apparatus comprising means for generating a 6-bit output code word in response to every 4-bit input code word by referring to a set of
25 encoding tables, wherein the encoding tables contain output code words assigned to input code words, and contain encoding-table

designation information accompanying each output code word,
wherein the encoding-table designation information designates an
encoding table among the encoding tables which is used next to
generate an output code word immediately following the output
5 code word accompanied with the encoding-table designation
information; and means for sequentially connecting the generated
output code words into a sequence of the generated output code
words which follows predetermined run length limiting rules (1,
k)RLL, where "k" denotes a predetermined natural number between
10 7 and 12.

A seventh aspect of this invention is based on the sixth aspect
thereof, and provides a modulation apparatus wherein NRZI
conversion results of output code words in first specified one of the
encoding tables which are assigned to prescribed input code words
15 are opposite in polarity to NRZI conversion results of output code
words in second specified one of the encoding tables which are
assigned to the prescribed input code words, and further
comprising means for generating a first candidate current output
code word in response to a current input code word equal to one of
20 the prescribed input code words by referring to the first specified
one of the encoding tables, and means for generating a second
candidate current output code word in response to the current
input code word equal to said one of the prescribed input code
words by referring to the second specified one of the encoding
25 tables, wherein a succession of a specified immediately-preceding
output code word and the first candidate current output code word

and also a succession of the specified immediately-preceding output code word and the second candidate current output code follow the predetermined run length limiting rules (1, k)RLL.

An eighth aspect of this invention is based on the seventh
5 aspect thereof, and provides a modulation apparatus further comprising means for selecting one from the first and second candidate current output code words as a final current output code word.

A ninth aspect of this invention is based on the seventh aspect
10 thereof, and provides a modulation apparatus further comprising means for calculating a first CDS of the first candidate current output code word; means for updating a first DSV of the first candidate current output code word and previous final output code words in response to the first CDS; means for calculating a second
15 CDS of the second candidate current output code word; means for updating a second DSV of the second candidate current output code word and previous final output code words in response to the second CDS; means for determining which of an absolute value of the first DSV and an absolute value of the second DSV is smaller;
20 and means for selecting one from the first and second candidate current output code words which corresponds to the smaller DSV absolute value as a final current output code word.

A tenth aspect of this invention is based on the sixth aspect thereof, and provides a modulation apparatus further comprising
25 means for predicting repetition of a minimum run length at least a predetermined number of times in the sequence of the generated

output code words; and means for, when the repetition of the minimum run length is predicted, changing an output code word causing the repetition to prevent the repetition of the minimum run length from occurring in the sequence of the generated output code words.

5 An eleventh aspect of this invention provides a demodulation method of demodulating a sequence of 6-bit code words which is generated by the modulation method in the first aspect of this invention. The demodulation method comprises the steps of
10 recovering encoding-table designation information from the code-word sequence, the encoding-table designation information representing which of encoding tables has been used in generating a code word immediately following a code word of interest; and demodulating the code word of interest into an original code word
15 by referring to a decoding table in response to the recovered encoding-table designation information.

A twelfth aspect of this invention provides a demodulation apparatus for demodulating a sequence of 6-bit code words which is generated by the modulation apparatus in the sixth aspect of this
20 invention. The demodulation apparatus comprises means for recovering encoding-table designation information from the code-word sequence, the encoding-table designation information representing which of encoding tables has been used in generating a code word immediately following a code word of interest; and
25 means for demodulating the code word of interest into an original code word by referring to a decoding table in response to the

recovered encoding-table designation information.

A thirteenth aspect of this invention provides an information recording medium storing a sequence of code words which is generated by the modulation apparatus in the sixth aspect of this invention.

A fourteenth aspect of this invention provides an information transmission method of transmitting a sequence of code words which is generated by the modulation method in the first aspect of this invention.

A fifteenth aspect of this invention provides an information transmission apparatus for transmitting a sequence of code words which is generated by the modulation apparatus in the sixth aspect of this invention.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a diagram of 6-bit output code words which follow (1, 7)RLL or (1, 8)RLL.

Fig. 2 is a diagram of an encoding table for converting every 4-bit input code word into a 6-bit output code word which is used in a modulation apparatus according to a first embodiment of this invention.

Fig. 3 is a diagram of another encoding table which can be used instead of the encoding table in Fig. 2.

Fig. 4 is a block diagram of the modulation apparatus according to the first embodiment of this invention.

Fig. 5 is a block diagram of a 4-6 modulator in Fig. 4.

Fig. 6 is a diagram of an example of five successive input code

words $D(k)$, five successive current-table selection numbers $S(k)$ for designating sub encoding tables used in encoding the input code words $D(k)$, five successive output code words $C(k)$ assigned to the input code words $D(k)$, and five successive next-table selection numbers $S(k+1)$ accompanying the output code words $C(k)$.

Fig. 7 is a flowchart of a segment of a control program for a code-word selection detector in Fig. 5.

Fig. 8 is a time-domain diagram of a succession of output code words $C(k-1)$, $C(k)0$, and $C(k+1)$ being "010000", "101001", and "000001", and the result of NRZI conversion of the output code words $C(k-1)$, $C(k)0$, and $C(k+1)$.

Fig. 9 is a time-domain diagram of a succession of output code words $C(k-1)$, $C(k)1$, and $C(k+1)$ being "010000", "001001", and "000001", and the result of NRZI conversion of the output code words $C(k-1)$, $C(k)1$, and $C(k+1)$.

Fig. 10 is a flowchart of a segment of a control program for the code-word selection detector in Fig. 5 which can replace the program segment in Fig. 7.

Fig. 11 is a flowchart of a segment of a control program for the 4-6 modulator in Fig. 4.

Fig. 12 is a block diagram of the code-word selection detector and a basic encoder in Fig. 5.

Fig. 13 is a block diagram of a demodulation apparatus according to a second embodiment of this invention.

Fig. 14 is a diagram of an example of the contents of a decoding table used in the demodulation apparatus of Fig. 13.

Fig. 15 is a diagram of a succession of input code words $C(k)$ being "010000", "001001", "000001", "000101", and "010001", a succession of reproduced original code words $D(k)$ corresponding to the input code words $C(k)$, a succession of states of decision information corresponding to the input code words $C(k)$, and a succession of encoding states $S(k)$ corresponding to the input code words $C(k)$.

DETAILED DESCRIPTION OF THE INVENTION

First Embodiment

10 Run length limiting rules "(d, k)RLL" are such that "d" to "k" successive bits of "0" should be between bits of "1" in a modulation-resultant bit stream, where "d" and "k" denote predetermined natural numbers and the number "d" is smaller than the number "k".

Fig. 1 shows 6-bit output code words which follow (1, 7)RLL or (1, 8)RLL. Fig. 2 shows an encoding table for converting or
15 encoding every 4-bit input code word (every 4-bit input data word) into a 6-bit output code word. The encoding table in Fig. 2 uses 6-bit output code words listed in Fig. 1.

The encoding table in Fig. 2 has a set of four sub encoding
20 tables having identification (ID) numbers of "0", "1", "2", and "3" respectively. Each of the four sub encoding tables stores 6-bit output code words $C(k)$ assigned to 4-bit input code words $D(k)$. The four sub encoding tables contain arrays of cells at different addresses respectively. Each of the cells has a set of an input code
25 word $D(k)$, an output code word $C(k)$ assigned to the input code word $D(k)$, and a number $S(k+1)$ assigned to the output code word

C(k). In Fig. 2, each input code word D(k) is expressed by the decimal notation while each output code word C(k) is expressed by both the decimal notation and the binary notation. In Fig. 2, each output code word C(k) is followed by and accompanied with a
5 number S(k+1) which designates a sub encoding table used next. Under normal conditions, when the number S(k+1) accompanying the current output code word is "0", the sub encoding table having an ID number of "0" is used to generate a next output code word. When the number S(k+1) accompanying the current output code
10 word is "1", the sub encoding table having an ID number of "1" is used to generate a next output code word. When the number S(k+1) accompanying the current output code word is "2", the sub encoding table having an ID number of "2" is used to generate a next output code word. When the number S(k+1) accompanying the
15 current output code word is "3", the sub encoding table having an ID number of "3" is used to generate a next output code word. The numbers S(k+1) are referred to as the next-table selection numbers S(k+1). The next-table selection numbers S(k+1) are designed so that a sequence of selected output code words will follow (1, 7)RLL
20 or (1, 8)RLL. A next-table selection number accompanying an output code word C(k-1) immediately preceding the current output code word C(k) is defined as a current-table selection number S(k) used for generation of the current output code word C(k).

The sub encoding table having an ID number of "1" and the sub
25 encoding table having an ID number of "2" are in a predetermined relation as follows. The NRZI modulation results (the NRZI

conversion results) of output code words assigned to prescribed input code words in the sub encoding table having an ID number of "1" are opposite in polarity ("odd-even" in the number of "1", that is, DSV-related polarity) to those of output code words in the sub encoding table having an ID number of "2". The opposite polarities cause a DSV (digital sum variation) in an increasing direction and a DSV in a decreasing direction, respectively. As mentioned later, in the case where the sub encoding table having an ID number of "2" is originally designated and a current input code word is identical with such a prescribed one, two output code words are read out from the sub encoding table having an ID number of "2" and the sub encoding table having an ID number of "1" as two candidate output code words respectively. In this case, one is selected from the two candidate output code words as a final output code word in response to DSV calculation results.

The sub encoding table having an ID number of "1" and the sub encoding table having an ID number of "3" are in a predetermined relation as follows. The NRZI modulation results of output code words assigned to prescribed input code words in the sub encoding table having an ID number of "1" are opposite in DSV-related polarity to those of output code words in the sub encoding table having an ID number of "3". The opposite polarities cause a DSV in an increasing direction and a DSV in a decreasing direction, respectively. As mentioned later, in the case where the sub encoding table having an ID number of "3" is originally designated and a current input code word is identical with such a prescribed

one, two output code words are read out from the sub encoding table having an ID number of "3" and the sub encoding table having an ID number of "1" as two candidate output code words respectively. In this case, one is selected from the two candidate
5 output code words as a final output code word in response to DSV calculation results.

The sub encoding table having an ID number of "0" and the sub encoding table having an ID number of "2" are in a predetermined relation as follows. The NRZI modulation results of output code
10 words assigned to prescribed input code words in the sub encoding table having an ID number of "0" are opposite in DSV-related polarity to those of output code words in the sub encoding table having an ID number of "2". The opposite polarities cause a DSV in an increasing direction and a DSV in a decreasing direction,
15 respectively. As mentioned later, in the case where the sub encoding table having an ID number of "2" is originally designated and a current input code word is identical with such a prescribed one, two output code words are read out from the sub encoding table having an ID number of "2" and the sub encoding table having
20 an ID number of "0" as two candidate output code words respectively. In this case, one is selected from the two candidate output code words as a final output code word in response to DSV calculation results.

In the four sub encoding tables of Fig. 2, each of some output
25 code words is assigned in common to a plurality of input code words, and the common output code words in the respective cells

are accompanied with different next-table selection numbers $S(k+1)$ respectively. This design is advantageous in reducing the volume of an encoding table. The assignment of next-table selection numbers $S(k+1)$ to output code words follows predetermined rules. Next-table selection numbers $S(k+1)$ accompanying transmitted output code words are not positively transmitted to a decoder side (a demodulation side). The decoder side utilizes the predetermined assignment rules, and thereby recovers a next-table selection number $S(k+1)$ accompanying a code word of interest and then uses the recovered next-table selection number $S(k+1)$ in decoding the code word of interest rather than decoding a code word immediately following the code word of interest. This design simplifies the decoding procedure.

Fig. 3 shows an encoding table which is similar to the encoding table in Fig. 2 except for assignment of output code words $C(k)$ to input code words $D(k)$. The encoding table in Fig. 3 may be used instead of the encoding table in Fig. 2.

The encoding table in Fig. 2 or Fig. 3 is designed for conversion of a 4-bit input code word into a 6-bit output code word. Since doubling a 4-bit input code word and a 6-bit output code word results in an 8-bit input code word and a 12-bit output code word, an encoding table for converting an 8-bit input code word into a 12-bit output code word can be made on the basis of the encoding table in Fig. 2 or Fig. 3. Accordingly, this invention contains 8-12 modulation in addition to 4-6 modulation.

Fig. 4 shows a modulation apparatus 1 according to a first

embodiment of this invention. As shown in Fig. 4, the modulation apparatus 1 includes a formatter 11, a 4-6 modulator 12, an NRZI (non-return-to-zero invert) converter 14, and a recording and driving circuit 15 which are sequentially connected in that order.

5 The formatter 11 receives a digital information signal (an input digital signal). The input digital signal represents information such as video information, audio information, or audio visual information. The formatter 11 adds an error correction code signal to the received digital information signal, and sectors and makes
10 the addition-resultant signal into a second digital signal of a predetermined control format conforming with a recording format used by a recording medium 2. The formatter 11 outputs the second digital signal to the 4-6 modulator 12. The second digital signal is also referred to as the source code signal. The source code
15 signal has a sequence of 4-bit input code words.

 The 4-6 modulator 12 includes an encoding table 13 using the encoding table in Fig. 2. Alternatively, the encoding table 13 may use the encoding table in Fig. 3. The 4-6 modulator 12 subjects the second digital signal (the source code signal) to 4-6 modulation by
20 referring to the encoding table 13. Thereby, the 4-6 modulator 12 converts the second digital signal into a third digital signal. In addition, the 4-6 modulator 12 repetitively adds a sync word to the third digital signal. The 4-6 modulator 12 outputs the third digital signal to the NRZI converter 14.

25 The NRZI converter 14 subjects the third digital signal (the output digital signal from the 4-6 modulator 12) to NRZI

modulation, thereby converting the third digital signal into a fourth digital signal which is of an NRZI code. The NRZI converter 14 outputs the fourth digital signal to the recording and driving circuit 15. The recording and driving circuit 15 records the fourth digital signal (the output digital signal from the NRZI converter 14) on a recording medium 2 via a recording head.

The fourth digital signal can be fed to a transmission encoder 31 from the recording and driving circuit 15. The device 31 encodes the fourth digital signal into a fifth digital signal which is of a code suited for transmission. The transmission encoder 31 outputs the fifth digital signal to a transmission medium 3. The fifth digital signal propagates along the transmission medium 3.

As shown in Fig. 5, the 4-6 modulator 12 includes two memories 124 and 125 in paths "0" and "1" respectively. The path memories 124 and 125 are also referred to as the code word memories. The 4-6 modulator 12 further includes a code-word selection detector 121 and a basic encoder 122. The code-word selection detector 121 is connected with the basic encoder 122. The basic encoder 122 is connected with the path memories 124 and 125.

The basic encoder 122 receives the source code signal from the formatter 11. The basic encoder 122 handles every 4-bit block of the source code signal as an input code word. The basic encoder 122 includes the encoding table 13 used for converting or encoding every 4-bit input code word into a 6-bit output code word. The basic encoder 122 also includes an address generator for producing

an address signal in response to every 4-bit input code word. The address signal designates one of the cells in the encoding table 13 which should be accessed.

The 4-6 modulator 12 further includes DSV circuits 126 and 127, a comparator 128, and a controller 129. The DSV circuit 126 is connected with the path memory 124, the comparator 128, and the controller 129. The DSV circuit 127 is connected with the path memory 125, the comparator 128, and the controller 129. The comparator 128 is connected with the code-word selection detector 121 and the controller 129. The controller 129 is connected with the path memories 124 and 125. The controller 129 is followed by the NRZI converter 14 (see Fig. 4).

The 4-6 modulator 12 operates as follows. The basic encoder 122 receives the source code signal from the formatter 11. The basic encoder 122 handles every 4-bit block of the source code signal as an input code word $D(k)$. In addition, the basic encoder 122 implements frame-by-frame signal processing. Here, "frame" means a sync frame corresponding to each prescribed segment of the source code signal. A given number of sync frames compose one recording sector. The basic encoder 122 has an initial table in addition to the encoding table 13. The initial table contains a predetermined sync word (a predetermined sync bit pattern) and a predetermined initial value of an adopted next-table selection number $S(k+1)$. During a start of every frame, the basic encoder 122 accesses the initial table, and reads out the sync word and the initial value therefrom. The basic encoder 122 outputs the read-out

sync word to the next stage, that is, the path memories 124 and 125. The basic encoder 122 stores the read-out sync word into the path memories 124 and 125. The basic encoder 122 sets the adopted next-table selection number $S(k+1)$ to the read-out initial value. The basic encoder 122 delays a signal representative of the adopted next-table selection number $S(k+1)$ by a time interval corresponding to one word, thereby generating a signal representative of a current-table selection number $S(k)$ which accompanies an immediately-previous output code word $C(k-1)$ under normal conditions. The basic encoder 122 outputs the signal of the current-table selection number $S(k)$ to the code-word selection detector 121. The code-word selection detector 121 receives the source code signal from the formatter 11. The code-word selection detector 121 handles every 4-bit block of the source code signal as a current input code word $D(k)$. The code-word selection detector 121 receives the signal of the current-table selection number $S(k)$ from the basic encoder 122. First, the current-table section number $S(k)$ is equal to the initial value. In addition, the code-word selection detector 121 is informed by the controller 129 of a latest output code word $C(k-1)$ which has been finally selected and decided. The code-word selection detector 121 detects whether or not an output code word corresponding to the current input code word $D(k)$ is uniquely decided, that is, whether or not selecting one from candidate output code words as a final output code word corresponding to the current input code word $D(k)$ is required on the basis of the current input code word $D(k)$,

the current-table selection number $S(k)$, and the latest selected output code word $C(k-1)$. The code-word selection detector 121 outputs a signal representative of a result of the detection to the basic encoder 122 and the comparator 128. In more detail, the code-word selection detector 121 decides whether or not the current input code word $D(k)$, the current-table selection number $S(k)$, and the latest selected output code word $C(k-1)$ are in prescribed conditions. When the current input code word $D(k)$, the current-table selection number $S(k)$, and the latest selected output code word $C(k-1)$ are in the prescribed conditions, the code-word selection detector 121 outputs a detection-result signal (a code-word selection signal) indicating that code-word selection is required. Otherwise, the code-word selection detector 121 outputs a detection-result signal (a code-word non-selection signal) indicating that code-word selection is not required.

In the case where the detection-result signal outputted from the code-word selection detector 121 indicates that code-word selection is required, the basic encoder 122 takes two candidate output code words $C(k)0$ and $C(k)1$ for the current input code word $D(k)$. Specifically, the basic encoder 122 generates two different addresses in response to the current input code word $D(k)$ and the current-table selection number $S(k)$, and accesses two of the four sub encoding tables in response to the generated addresses. One of the two accessed sub encoding tables has an ID number equal to the current-table selection number $S(k)$. The basic encoder 122 reads out an output code word $C(k)0$ assigned to the current input code

word $D(k)$ from the sub encoding table having an ID number equal to the current-table selection number $S(k)$. The read-out output code word $C(k)0$ is defined as the first candidate output code word $C(k)0$. The basic encoder 122 reads out an output code word $C(k)1$

5 assigned to the current input code word $D(k)$ from the other accessed sub encoding table. The read-out output code word $C(k)1$ is defined as the second candidate output code word $C(k)1$. The candidate output code words $C(k)0$ and $C(k)1$ are assigned to the path "0" and the path "1", respectively. The basic encoder 122
10 stores the candidate output code words $C(k)0$ and $C(k)1$ into the path memories 124 and 125, respectively.

In the case where the detection-result signal outputted from the code-word selection detector 121 indicates that code-word selection is not required, the basic encoder 122 takes only one
15 output code word $C(k)$ for the current input code word $D(k)$. Specifically, the basic encoder 122 generates only one address in response to the current input code word $D(k)$ and the current-table selection number $S(k)$, and accesses one of the four sub encoding tables in response to the generated address. The accessed sub
20 encoding table has an ID number equal to the current-table selection number $S(k)$. The basic encoder 122 reads out an output code word $C(k)$ assigned to the current input code word $D(k)$ from the sub encoding table having an ID number equal to the current-table selection number $S(k)$. The basic encoder 122 stores the
25 output code word $C(k)$ into the path memory 124 as a first candidate output code word $C(k)0$. The basic encoder 122 stores

the output code word $C(k)$ into the path memory 125 as a second candidate output code word $C(k)1$. In this way, the same output code word $C(k)$ is written into the path memories 124 and 125. The basic encoder 122 updates the adopted next-table selection
5 number $S(k+1)$ to the value accompanying the output code word $C(k)$.

The DSV circuit 126 calculates a CDS (code digital sum) value of the output code word $C(k)0$ in the path memory 124, and updates a DSV value of the output code word $C(k)0$ and previous output code
10 words in response to the calculated CDS value. The DSV circuit 126 has a memory loaded with a signal representative of the updating-resultant DSV value (the newest DSV value). The DSV value provided by the DSV circuit 126 relates to the path "0". Similarly, the DSV circuit 127 calculates a CDS (code digital sum) value of the
15 output code word $C(k)1$ in the path memory 125, and updates a DSV value of the output code word $C(k)$ and previous output code words in response to the calculated CDS value. The DSV circuit 127 has a memory loaded with a signal representative of the updating-resultant DSV value (the newest DSV value). The DSV value
20 provided by the DSV circuit 127 relates to the path "1".

The comparator 128 responds to the detection-result signal outputted from the code-word selection detector 121. In the case where the detection-result signal indicates that code-word selection is required, the comparator 128 accesses the memories within the
25 DSV circuits 126 and 127. The comparator 128 calculates the absolute newest DSV value (the first absolute DSV value) stored in

the memory within the DSV circuit 126. The comparator 128 calculates the absolute newest DSV value (the second absolute DSV value) stored in the memory within the DSV circuit 127. The device 128 compares the first and second absolute DSV values to
5 decide which of the two is smaller. The comparator 128 notifies the result of the comparison to the controller 129. In the case where the detection-result signal indicates that code-word selection is not required, the comparator 128 is inactive and does not notify any comparison result to the controller 129.

10 When the comparison result notified by the comparator 128 indicates that the first absolute DSV value is smaller than the second absolute DSV value, the controller 129 reads out the output code word $C(k)0$ from the path memory 124. The controller 129 transmits the read-out output code word $C(k)0$ to the NRZI
15 converter 14 as a finally-selected output code word. The controller 129 informs the code-word selection detector 121 of the read-out output code word as the latest selected output code word $C(k-1)$. In addition, the controller 129 replaces the contents of the output code word $C(k)1$ in the path memory 125 with the contents of the
20 output code word $C(k)0$. Thus, in this case, the contents of the output code word $C(k)1$ in the path memory 125 are updated to the contents of the output code word $C(k)0$ in the path memory 124. Furthermore, the controller 129 reads out the DSV value from the memory within the DSV circuit 126, and updates the DSV value in
25 the memory within the DSV circuit 127 to the read-out DSV value. Thus, in this case, the DSV value in the memory within the DSV

circuit 127 is set to the DSV value in the memory within the DSV circuit 126. In addition, the controller 129 informs the basic encoder 122 that the output code word $C(k)0$ has been selected. The basic encoder 122 updates the adopted next-table selection number $S(k+1)$ to the value accompanying the output code word $C(k)0$.

When the comparison result notified by the comparator 128 indicates that the first absolute DSV value is equal to or greater than the second absolute DSV value, the controller 129 reads out the output code word $C(k)1$ from the path memory 125. The controller 129 transmits the read-out output code word $C(k)1$ to the NRZI converter 14 as a finally-selected output code word. The controller 129 informs the code-word selection detector 121 of the read-out output code word as the latest selected output code word $C(k-1)$. In addition, the controller 129 replaces the contents of the output code word $C(k)0$ in the path memory 124 with the contents of the output code word $C(k)1$. Thus, in this case, the contents of the output code word $C(k)0$ in the path memory 124 are updated to the contents of the output code word $C(k)1$ in the path memory 125. Furthermore, the controller 129 reads out the DSV value from the memory within the DSV circuit 127, and updates the DSV value in the memory within the DSV circuit 126 to the read-out DSV value. Thus, in this case, the DSV value in the memory within the DSV circuit 126 is set to the DSV value in the memory within the DSV circuit 127. In addition, the controller 129 informs the basic encoder 122 that the output code word $C(k)1$ has been selected.

The basic encoder 122 updates the adopted next-table selection number $S(k+1)$ to the value accompanying the output code word $C(k)1$.

In the absence of the comparison result notified by the
5 comparator 128, the controller 129 reads out the output code word $C(k)0$ from the path memory 124. The controller 129 transmits the read-out output code word $C(k)0$ to the NRZI converter 14 as a finally-selected output code word. The controller 129 informs the code-word selection detector 121 of the read-out output code word
10 as the latest selected output code word $C(k-1)$. In this case, the controller 129 does not access the path memory 125 and the DSV circuits 126 and 127.

It should be noted that the number of candidate output code words may be three or more. In this case, one of the candidate
15 output code words which corresponds to the smallest DSV value is selected as a final output code word. First and second sequences of output code words corresponding to all input code words may be stored in the path memories 124 and 125. In this case, after an end input code word has been modulated, the controller 129
20 selects one from the first and second sequences of output code words in the path memories 124 and 125 and transmits the selected sequence to the NRZI converter 14.

Fig. 6 shows an example of five successive input code words. With reference to Fig. 6, there is a sequence of input code words of
25 "4", "5", "6", "7", and "8" (decimal). At an initial stage, the current-table selection number $S(k)$ is set to an initial value of, for example,

"0". Thus, the sub encoding table having an ID number of "0" is accessed for the first input code word "4", and an output code word of "18" (decimal) equal to "010010" (binary) which is assigned to the first input code word "4" is read out from the accessed sub encoding table (see Fig. 2). The bit sequence "010010" is outputted. At the same time, a number $S(k+1)$ of "1" which accompanies the output code word "010010" is read out from the accessed sub encoding table. Then, the current-table selection number $S(k)$ is updated to the read-out value "1". Thus, the sub encoding table having an ID number of "1" is accessed for the second input code word "5", and an output code word of "2" (decimal) equal to "000010" (binary) which is assigned to the second input code word "5" is read out from the accessed sub encoding table (see Fig. 2). The bit sequence "000010" is outputted. At the same time, a number $S(k+1)$ of "2" which accompanies the output code word "000010" is read out from the accessed sub encoding table. Then, the current-table selection number $S(k)$ is updated to the read-out value "2". Thus, the sub encoding table having an ID number of "2" is accessed for the third input code word "6", and an output code word of "18" (decimal) equal to "010010" (binary) which is assigned to the third input code word "6" is read out from the accessed sub encoding table (see Fig. 2). The bit sequence "010010" is outputted. At the same time, a number $S(k+1)$ of "3" which accompanies the output code word "000010" is read out from the accessed sub encoding table. Then, the current-table selection number $S(k)$ is updated to the read-out value "3". Thus, the sub encoding table

having an ID number of "3" is accessed for the fourth input code word "7", and an output code word of "21" (decimal) equal to "010101" (binary) which is assigned to the fourth input code word "7" is read out from the accessed sub encoding table (see Fig. 2).

- 5 The bit sequence "010101" is outputted. At the same time, a number $S(k+1)$ of "0" which accompanies the output code word "010101" is read out from the accessed sub encoding table. Then, the current-table selection number $S(k)$ is updated to the read-out value "0". Thus, the sub encoding table having an ID number of "0"
- 10 is accessed for the fifth input code word "8", and an output code word of "21" (decimal) equal to "010101" (binary) which is assigned to the fifth input code word "8" is read out from the accessed sub encoding table (see Fig. 2). The bit sequence "010101" is outputted. At the same time, a number $S(k+1)$ of "1" which accompanies the
- 15 output code word "010101" is read out from the accessed sub encoding table. Then, the current-table selection number $S(k)$ is updated to the read-out value "1".

In this way, a sequence of input code words of "4", "5", "6", "7", and "8" is converted into a sequence of output code words as

20 "010010", "000010", "010010", "010101", and "010101". A bit stream formed by sequentially direct connection of the output code words is "0100100000100100100101010101". This bit stream follows (1, 7)RLL.

The code-word selection detector 121 may be formed by a

25 digital signal processor, a CPU, or a similar device including a combination of an input/output port, a processing section, a ROM,

and a RAM. In this case, the code-word selection detector 121 operates in accordance with a control program stored in the ROM.

Fig. 7 is a flowchart of a segment of the control program for the code-word selection detector 121 which is executed for every input code word. With reference to Fig. 7, a first step 201 of the program segment detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The latest selected output code word $C(k-1)$ is fed from the controller 129. The step 201 decides which of predetermined values the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to. When the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "4", that is, when the latest selected output code word $C(k-1)$ is "010000", the program advances from the step 201 to a step 202. When the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "5", that is, when the latest selected output code word $C(k-1)$ is "100000", the program advances from the step 201 to a step 209. When the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to neither "4" nor "5", the program advances from the step 201 to a step 205.

The step 202 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The current-table selection number $S(k)$ is notified by the basic encoder 122. The step 202 decides whether or not the current-table selection number $S(k)$ is "3" and the current input code word $D(k)$ is less than "4" (decimal). In other words, the step 202 decides whether

or not the current-table selection number $S(k)$ is "3" and the current input code word $D(k)$ is in the range of "0" to "3" (decimal). When the current-table selection number $S(k)$ is "3" and the current input code word $D(k)$ is in the range of "0" to "3", the program
5 advances from the step 202 to a step 206. Otherwise, the program advances from the step 202 to a step 203.

The step 203 decides whether or not the current-table selection number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "6" (decimal). When the current table-table selection
10 number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "6", the program advances from the step 203 to a step 207. Otherwise, the program advances from the step 202 to a step 208.

The step 209 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 209 decides
15 whether or not the current-table selection number $S(k)$ is "3" and the current input code word $D(k)$ is less than "2" (decimal). In other words, the step 209 decides whether or not the current-table selection number $S(k)$ is "3" and the current input code word $D(k)$ is in the range of "0" to "1" (decimal). When the current-table
20 selection number $S(k)$ is "3" and the current input code word $D(k)$ is in the range of "0" to "1", the program advances from the step 209 to a step 210. Otherwise, the program advances from the step 209 to a step 211.

The step 211 decides whether or not the current-table
25 selection number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "9" (decimal). When the current-table selection

number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "9", the program advances from the step 211 to a step 212. Otherwise, the program advances from the step 211 to the step 208.

- 5 The step 205 detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The step 205 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 205 decides whether or not all the following conditions A1, A2, and A3 are satisfied. A1: The detected LSB-side
10 zero run length of the latest selected output code word $C(k-1)$ is equal to "1" or "2". In other words, the latest selected output code word $C(k-1)$ is "010100", "000100", "100100", "010010", "000010", "001010", "101010", or "100010". A2: The current-table selection number $S(k)$ is "2". A3: The current input code word $D(k)$
15 is less than "2" (decimal). In other words, the current input code word $D(k)$ is in the range of "0" to "1" (decimal). When all the conditions A1, A2, and A3 are satisfied, the program advances from the step 205 to a step 214. Otherwise, the program advances from the step 205 to a step 215.
- 20 The step 215 detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The step 215 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 215 decides whether or not all the following conditions B1, B2, and B3 are satisfied. B1: The detected LSB-side
25 zero run length of the latest selected output code word $C(k-1)$ is equal to "1". In other words, the latest selected output code word

$C(k-1)$ is "010010", "000010", "001010", "101010", or "100010".

B2: The current-table selection number $S(k)$ is "2". B3: The current input code word $D(k)$ is in the range of "12 to "13" (decimal). When all the conditions B1, B2, and B3 are satisfied, the program

5 advances from the step 215 to a step 217. Otherwise, the program advances from the step 215 to the step 208.

The step 217 determines an output code word $C(k+1)$ assigned to a next input code word $D(k+1)$, that is, an input code word $D(k+1)$ immediately following the current input code word

10 $D(k)$. Specifically, the step 217 reads the next input code word $D(k+1)$. The step 217 determines an output code word $C(k)$ immediately following the latest selected output code word $C(k-1)$ in response to the current input code word $D(k)$ by referring to the sub encoding table in the basic encoder 122 which has an ID

15 number of "0" or "2". The step 217 reads out a next-table selection number $S(k+1)$ accompanying the determined output code word $C(k)$ from the accessed sub encoding table. The step 217 reads out an output code word $C(k+1)$ assigned to the next input code word $D(k+1)$ from the sub encoding table having an ID number equal to

20 the read-out next-table selection number $S(k+1)$. Thereafter, the step 217 decides whether or not the MSB of the read-out output code word $C(k+1)$ is "1". When the MSB of the read-out output code word $C(k+1)$ is "1", the program advances from the step 217 to a step 218. Otherwise, the program advances from the step 217 to

25 the step 208.

The step 206 generates a code-word selection signal designed

for using the sub encoding table in the basic encoder 122 which has an ID number of "3" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "1" to generate a second candidate

5 output code word $C(k)1$. The step 206 outputs the generated code-word selection signal. After the step 206, the current execution cycle of the program segment ends.

The step 207 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID
10 number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "1" to generate a second candidate output code word $C(k)1$. The step 207 outputs the generated code-word selection signal. After the step 207, the current execution cycle of
15 the program segment ends.

The step 210 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "3" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which
20 has an ID number of "1" to generate a second candidate output code word $C(k)1$. The step 210 outputs the generated code-word selection signal. After the step 210, the current execution cycle of the program segment ends.

The step 212 generates a code-word selection signal for using
25 the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$,

and for using the sub encoding table in the basic encoder 122 which has an ID number of "1" to generate a second candidate output code word $C(k)1$. The step 212 outputs the generated code-word selection signal. After the step 212, the current execution cycle of the program segment ends.

The step 214 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "0" to generate a second candidate output code word $C(k)1$. The step 214 outputs the generated code-word selection signal. After the step 214, the current execution cycle of the program segment ends.

The step 218 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "0" to generate a second candidate output code word $C(k)1$. The step 218 outputs the generated code-word selection signal. After the step 218, the current execution cycle of the program segment ends.

The step 208 generates a code-word non-selection signal. The step 208 outputs the generated code-word non-selection signal. After the step 208, the current execution cycle of the program segment ends.

In the case where the latest selected output code word $C(k-1)$

is "010000" and the current-table selection number $S(k)$ is "3", and where the current input code word $D(k)$ is in the range of "0" to "3" (decimal), when the originally-designated sub encoding table having an ID number of "3" is used to generate an output code word $C(k)$, a
5 resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. In this case, even when the sub encoding table having an ID number of "1" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$
10 follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the sub encoding table having an ID number of "2" or "3" will be used to generate an output code word $C(k)$ immediately following the output code word $C(k-1)$ being "010000". In the sub encoding tables having ID numbers of "1", "2", and "3", output code words assigned
15 to a same input code word are different from each other. Therefore, using the sub encoding table having an ID number of "1" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the combination of the steps 201, 202, and 206.
20 In the case where the latest selected output code word $C(k-1)$ is "010000" and the current-table selection number $S(k)$ is "2", and where the current input code word $D(k)$ is greater than "6" (decimal), when the originally-designated sub encoding table having an ID number of "2" is used to generate an output code word $C(k)$, a
25 resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. In this case, even when the sub encoding table

having an ID number of "1" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the
5 sub encoding table having an ID number of "2" or "3" will be used to generate an output code word $C(k)$ immediately following the output code word $C(k-1)$ being "010000". In the sub encoding tables having ID numbers of "1", "2", and "3", output code words assigned to a same input code word are different from each other.
10 Therefore, using the sub encoding table having an ID number of "1" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the combination of the steps 201, 203, and 207.

In the case where the latest selected output code word $C(k-1)$
15 is "100000" and the current-table selection number $S(k)$ is "3", and where the current input code word $D(k)$ is in the range of "0" to "1" (decimal), when the originally-designated sub encoding table having an ID number of "3" is used to generate an output code word $C(k)$, a resultant succession of the output code words $C(k-1)$ and $C(k)$
20 follows (1, 7) RLL. In this case, even when the sub encoding table having an ID number of "1" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the
25 sub encoding table having an ID number of "2" or "3" will be used to generate an output code word $C(k)$ immediately following the output

code word $C(k-1)$ being "100000". In the sub encoding tables having ID numbers of "1", "2", and "3", output code words assigned to a same input code word are different from each other.

Therefore, using the sub encoding table having an ID number of "1" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the combination of the steps 201, 209, and 210.

In the case where the latest selected output code word $C(k-1)$ is "100000" and the current-table selection number $S(k)$ is "2", and where the current input code word $D(k)$ is greater than "9" (decimal), when the originally-designated sub encoding table having an ID number of "2" is used to generate an output code word $C(k)$, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. In this case, even when the sub encoding table having an ID number of "1" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the sub encoding table having an ID number of "2" or "3" will be used to generate an output code word $C(k)$ immediately following the output code word $C(k-1)$ being "100000". In the sub encoding tables having ID numbers of "1", "2", and "3", output code words assigned to a same input code word are different from each other. Therefore, using the sub encoding table having an ID number of "1" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the

combination of the steps 201, 211, and 212.

In the case where the latest selected output code word $C(k-1)$ has an LSB-side zero run length of "1" or "2" and the current-table selection number $S(k)$ is "2", and where the current input code word $D(k)$ is less than "2" (decimal), when the originally-designated sub encoding table having an ID number of "2" is used to generate an output code word $C(k)$, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. In this case, even when the sub encoding table having an ID number of "0" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the sub encoding table having an ID number of "1", "2", or "3" will be used to generate an output code word $C(k)$ immediately following the output code word $C(k-1)$ having an LSB-side zero run length of "1" or "2". In the sub encoding tables having ID numbers of "0", "1", "2", and "3", output code words assigned to a same input code word of "0" or "1" (decimal) are different from each other. Therefore, using the sub encoding table having an ID number of "0" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the combination of the steps 205 and 214.

In the case where the latest selected output code word $C(k-1)$ has an LSB-side zero run length of "1" and the current-table selection number $S(k)$ is "2", and where the current input code word $D(k)$ is "12" or "13" (decimal) and the MSB of the estimated

output code word $C(k+1)$ is "1", when the originally-designated sub encoding table having an ID number of "2" is used to generate an output code word $C(k)$, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. In this case, even when

5 the sub encoding table having an ID number of "0" is used to generate an output code word $C(k)$ instead of the originally-designated sub encoding table, a resultant succession of the output code words $C(k-1)$ and $C(k)$ follows (1, 7) RLL. The encoding table 13 in Fig. 2 shows that the sub encoding table having an ID number

10 of "1", "2", or "3" will be used to generate an output code word $C(k)$ immediately following the output code word $C(k-1)$ having an LSB-side zero run length of "1". In the sub encoding tables having ID numbers of "0", "1", "2", and "3", output code words assigned to a same input code word of "12" or "13" (decimal) are different from

15 each other. Therefore, using the sub encoding table having an ID number of "0" instead of the originally-designated sub encoding table will not cause a problem in a decoding side. This case corresponds to the combination of the steps 215, 217, and 218.

DSV control is implemented as follows. In the case where the

20 latest selected output code word $C(k-1)$ is "010000" and the current-table selection number $S(k)$ is "3", and where the current input code word $D(k)$ is "0" (decimal), the originally-designated sub encoding table having an ID number of "3" and also the sub encoding table having an ID number of "1" are accessed. Output code words

25 assigned to the current input code word $D(k)$ are read out from the accessed sub encoding tables. The output code word read out from

the sub coding table having an ID number of "3" is set as a first candidate output code word $C(k)0$. The output code word read out from the sub coding table having an ID number of "1" is set as a second candidate output code word $C(k)1$. The first candidate

5 output code word $C(k)0$ is "101001" while the second candidate output code word $C(k)1$ is "001001". It is assumed that a next output code word $C(k+1)$ is "000001". Fig. 8 shows a succession of the output code words $C(k-1)$, $C(k)0$, and $C(k+1)$, that is, "010000", "101001", and "000001". Fig. 8 also shows the result of NRZI

10 conversion of the output code words $C(k-1)$, $C(k)0$, and $C(k+1)$. Fig. 9 shows a succession of the output code words $C(k-1)$, $C(k)1$, and $C(k+1)$, that is, "010000", "001001", and "000001". Fig. 9 also shows the result of NRZI conversion of the output code words $C(k-1)$, $C(k)1$, and $C(k+1)$. As shown in Fig. 8, the result of NRZI

15 conversion of the first candidate output code word $C(k)0$ is "111000". As shown in Fig. 9, the result of NRZI conversion of the second candidate output code word $C(k)1$ is "001111". Therefore, the first and second candidate output code words $C(k)0$ and $C(k)1$ cause different DSV-related polarities regarding the NRZI

20 conversion results respectively. Thus, the first and second candidate output code words $C(k)0$ and $C(k)1$ cause different DSV values respectively. As previously mentioned, one of the first and second candidate output code words $C(k)0$ and $C(k)1$ which causes the smaller DSV value is selected as a final output code word $C(k)$.

25 The code-word selection provides DSV control of suppressing a DC component of a modulation-resultant bit stream.

It should be noted that (1, 7)RLL may be replaced by (1, 8)RLL. Fig. 10 is a flowchart of a segment of a control program for the code-word selection detector 121 which replaces the program segment in Fig. 7. The program segment in Fig. 10 is designed for
5 (1, 8)RLL.

With reference to Fig. 10, a first step 301 of the program segment detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The latest selected output code word $C(k-1)$ is fed from the controller 129. The step 301 decides
10 whether or not the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is in the range of "4" to "5". When the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is in the range of "4" to "5", the program advances from the step 301 to a step 302. Otherwise, the program
15 advances from the step 301 to a step 307.

The step 302 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The current-table selection number $S(k)$ is notified by the basic encoder 122. The step 302 decides whether or not all the following conditions C1, C2,
20 and C3 are satisfied. C1: The detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "4". C2: The current-table selection number $S(k)$ is "3". C3: The current input code word $D(k)$ is less than "7" (decimal). When all the conditions C1, C2, and C3 are satisfied, the program advances from the step
25 302 to a step 303. In addition, the step 302 decides whether or not all the following conditions D1, D2, and D3 are satisfied. D1:

The detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "5". D2: The current-table selection number $S(k)$ is "3". D3: The current input code word $D(k)$ is less than "4" (decimal). When all the conditions D1, D2, and D3 are
5 satisfied, the program advances from the step 302 to the step 303. In other cases, the program advances from the step 302 to a step 304.

The step 304 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 304 decides
10 whether or not the current-table selection number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "6" (decimal). When the current-table selection number $S(k)$ is "2" and the current input code word $D(k)$ is greater than "6", the program advances from the step 304 to a step 305. Otherwise, the program advances
15 from the step 304 to a step 306.

The step 307 detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The step 307 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 307 decides whether or not all the following
20 conditions E1, E2, and E3 are satisfied. E1: The detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "1". E2: The current-table selection number $S(k)$ is "2". E3: The current input code word $D(k)$ is "12" or "13" (decimal). When all the conditions E1, E2, and E3 are satisfied, the program
25 advances from the step 307 to a step 309. Otherwise, the program advances from the step 307 to a step 310.

The step 310 detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The step 310 checks the current input code word $D(k)$ and the current-table selection number $S(k)$. The step 310 decides whether or not all the following

5 conditions F1, F2, and F3 are satisfied. F1: The detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to or less than "3". F2: The current-table selection number $S(k)$ is "2". F3: The current input code word $D(k)$ is less than "2" (decimal). When all the conditions F1, F2, and F3 are satisfied, the

10 program advances from the step 310 to a step 312. Otherwise, the program advances from the step 310 to a step 313.

The step 313 detects the zero run length of the LSB side of the latest selected output code word $C(k-1)$. The step 313 checks the current input code word $D(k)$ and the current-table selection

15 number $S(k)$. The step 313 decides whether or not all the following conditions G1, G2, and G3 are satisfied. G1: The detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "2". G2: The current-table selection number $S(k)$ is "2". G3: The current input code word $D(k)$ is "12" or "13" (decimal).

20 When all the conditions G1, G2, and G3 are satisfied, the program advances from the step 313 to a step 315. Otherwise, the program advances from the step 313 to the step 306.

The step 315 determines an output code word $C(k+1)$ assigned to a next input code word $D(k+1)$, that is, an input code

25 word $D(k+1)$ immediately following the current input code word $D(k)$. Specifically, the step 315 reads the next input code word

D(k+1). The step 315 determines an output code word C(k) immediately following the latest selected output code word C(k-1) in response to the current input code word D(k) by referring to the sub encoding table in the basic encoder 122 which has an ID number of "0" or "2". The step 315 reads out a next-table selection number S(k+1) accompanying the determined output code word C(k) from the accessed sub encoding table. The step 315 reads out an output code word C(k+1) assigned to the next input code word D(k+1) from the sub encoding table having an ID number equal to the read-out next-table selection number S(k+1). Thereafter, the step 315 decides whether or not the MSB of the read-out output code word C(k+1) is "1". When the MSB of the read-out output code word C(k+1) is "1", the program advances from the step 315 to a step 316. Otherwise, the program advances from the step 315 to the step 306.

The step 303 generates a code-word selection signal designed for using the sub encoding table in the basic encoder 122 which has an ID number of "3" to generate a first candidate output code word C(k)0, and for using the sub encoding table in the basic encoder 122 which has an ID number of "1" to generate a second candidate output code word C(k)1. The step 303 outputs the generated code-word selection signal. After the step 303, the current execution cycle of the program segment ends.

The step 305 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word C(k)0,

and for using the sub encoding table in the basic encoder 122 which has an ID number of "1" to generate a second candidate output code word $C(k)1$. The step 305 outputs the generated code-word selection signal. After the step 305, the current execution cycle of the program segment ends.

The step 309 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "0" to generate a second candidate output code word $C(k)1$. The step 309 outputs the generated code-word selection signal. After the step 309, the current execution cycle of the program segment ends.

The step 312 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "0" to generate a second candidate output code word $C(k)1$. The step 312 outputs the generated code-word selection signal. After the step 312, the current execution cycle of the program segment ends.

The step 316 generates a code-word selection signal for using the sub encoding table in the basic encoder 122 which has an ID number of "2" to generate a first candidate output code word $C(k)0$, and for using the sub encoding table in the basic encoder 122 which has an ID number of "0" to generate a second candidate output code

word $C(k)1$. The step 316 outputs the generated code-word selection signal. After the step 316, the current execution cycle of the program segment ends.

The step 306 generates a code-word non-selection signal.

- 5 The step 306 outputs the generated code-word non-selection signal. After the step 306, the current execution cycle of the program segment ends.

10 In the case where $(1, 8)\text{RLL}$ is replaced by $(1, 9)\text{RLL}$, the program segment in Fig. 10 is modified as follows. The step 301 is modified to additionally decide whether or not the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is "6". When $k=9$ is satisfied, the step 303 or 305 is executed. Here, "k" denotes one in $(1, k)\text{RLL}$. The step 315 is removed. Thus, when the step 313 decides that all the conditions G1, G2, and
15 G3 are satisfied, the program advances from the step 313 to the step 316.

In the case where $(1, 8)\text{RLL}$ is replaced by $(1, 10)\text{RLL}$, the program segment in Fig. 10 is modified as follows. The step 301 is modified to additionally decide whether or not the detected LSB-
20 side zero run length of the latest selected output code word $C(k-1)$ is "6". The condition G1 used by the step 313 is modified to mean that the detected LSB-side zero run length of the latest selected output code word $C(k-1)$ is equal to "2" or "3". The step 315 is removed. Thus, when the step 313 decides that all the conditions
25 G1, G2, and G3 are satisfied, the program advances from the step 313 to the step 316.

In the case where (1, 10)RLL is replaced by (1, 11)RLL, the program segment is further modified as follows. The condition G1 used by the step 313 is modified to mean that the detected LSB-side zero run length of the latest selected output code word C(k-1) is equal to "2", "3", or "4". The step 315 is removed. Thus, when the step 313 decides that all the conditions G1, G2, and G3 are satisfied, the program advances from the step 313 to the step 316.

In the case where (1, 10)RLL is replaced by (1, 12)RLL, the program segment is further modified as follows. The condition G1 used by the step 313 is modified to mean that the detected LSB-side zero run length of the latest selected output code word C(k-1) is equal to "2", "3", "4", or "5". The step 315 is removed. Thus, when the step 313 decides that all the conditions G1, G2, and G3 are satisfied, the program advances from the step 313 to the step 316.

The 4-6 modulator 12 may be formed by a digital signal processor, a CPU, or a similar device including a combination of an input/output port, a processing section, a ROM, and a RAM. In this case, the 4-6 modulator 12 operates in accordance with a control program stored in the ROM. The encoding table 13 and the initial table are provided in the ROM while the path memories 124 and 125, and the memories within the DSV circuits 126 and 127 are provided in the RAM.

Fig. 11 is a flowchart of a segment of the control program for the 4-6 modulator 12. The program segment in Fig. 11 is executed for every sync frame. As shown in Fig. 11, a first step 101 of the

program segment reads out the initial value from the initial table.
The step 101 sets the current-table selection number $S(k)$ to the
read-out initial value. The step 101 initializes the DSV values (the
path-0 and path-1 DSV values). After the step 101, the program
5 advances to a step 102.

The step 102 receives a current input code word $D(k)$. A step
103 following the step 102 decides whether or not prescribed
conditions for code-word selection are satisfied, that is, whether or
not code-word selection should be implemented. The prescribed
10 conditions correspond to the conditions for code-word selection in
Fig. 7 (or Fig. 10). Thus, the prescribed conditions relate to the
detected LSB-side zero run length of a latest selected output code
word $C(k-1)$, the current-table selection number $S(k)$, the current
input code word $D(k)$, and the MSB of a next output code word
15 $C(k+1)$. When the prescribed conditions are satisfied, that is, when
code-word selection should be implemented, the program advances
from the step 103 to a step 104. Otherwise, the program advances
from the step 103 to a step 114.

The step 104 chooses two among the sub encoding tables
20 which should be accessed. A first sub encoding table to be accessed
has an ID number equal to the current-table selection number $S(k)$.
A second sub encoding table to be accessed has an ID number
determined by the prescribed conditions used in the step 103. The
step 104 reads out an output code word $C(k)0$ assigned to the
25 current input code word $D(k)$ from the first chosen sub encoding
table. The step 104 reads out an output code word $C(k)1$ assigned

to the current input code word $D(k)$ from the second chosen sub encoding table. The read-out output code word $C(k)0$ is defined as the first candidate output code word $C(k)0$ assigned to the path "0". The read-out output code word $C(k)1$ is defined as the second
5 candidate output code word $C(k)1$ assigned to the path "1".

A step 105 following the step 104 calculates a CDS value of the first candidate output code word $C(k)0$, and updates the path-0 DSV value of the first candidate output code word $C(k)0$ and previous
10 output code words in response to the calculated CDS value. In addition, the step 105 calculates a CDS value of the second candidate output code word $C(k)1$, and updates the path-1 DSV value of the second candidate output code word $C(k)1$ and previous output code words in response to the calculated CDS value.

A step 106 subsequent to the step 105 calculates the absolute
15 path-0 DSV value and the absolute path-1 DSV value. The step 106 compares the absolute path-0 DSV value and the absolute path-1 DSV value to decide which of the two is smaller. When the absolute path-0 DSV value is smaller than the absolute path-1 DSV value, the step 106 outputs the first candidate output code word $C(k)0$ as a
20 finally-selected output code word. In addition, the step 106 replaces the contents of the second output code word $C(k)1$ with the contents of the first output code word $C(k)0$. Furthermore, the step 106 equalizes the path-1 DSV value to the path-0 DSV value. Also, the step 106 sets the current-table selection number $S(k)$ to
25 the value accompanying the first candidate output code word $C(k)0$. On the other hand, when the absolute path-0 DSV value is equal to

or greater than the absolute path-1 DSV value, the step 106 outputs the second candidate output code word $C(k)1$ as a finally-selected output code word. In addition, the step 106 replaces the contents of the first output code word $C(k)0$ with the contents of the second
5 output code word $C(k)1$. Furthermore, the step 106 equalizes the path-0 DSV value to the path-1 DSV value. Also, the step 106 sets the current-table selection number $S(k)$ to the value accompanying the second candidate output code word $C(k)1$. After the step 106, the program advances to a step 107.

10 The step 114 accesses the sub encoding table having an ID number equal to the current-table selection number $S(k)$. The step 114 reads out an output code word $C(k)$ assigned to the current input code word $D(k)$ from the accessed sub encoding table. The read-out output code word $C(k)$ is defined as the first candidate
15 output code word $C(k)0$ assigned to the path "0" and also the second candidate output code word $C(k)1$ assigned to the path "1".

A step 115 following the step 114 calculates a CDS value of the first candidate output code word $C(k)0$, and updates the path-0 DSV value of the first candidate output code word $C(k)0$ and previous
20 output code words in response to the calculated CDS value. In addition, the step 115 calculates a CDS value of the second candidate output code word $C(k)1$, and updates the path-1 DSV value of the second candidate output code word $C(k)1$ and previous output code words in response to the calculated CDS value.

25 A step 116 subsequent to the step 115 outputs the first candidate output code word $C(k)0$ as a finally-selected output code

word. In addition, the step 116 sets the current-table selection number $S(k)$ to the value accompanying the first candidate output code word $C(k)0$. After the step 116, the program advances to the step 107.

5 The step 107 decides whether or not the current input code word $D(k)$ corresponds to an end of a frame. When the current input code word $D(k)$ corresponds to an end of a frame, the program exits from the step 107 and then the current execution cycle of the program segment ends. Otherwise, the program
10 returns from the step 107 to the step 102.

 In the case of a transmission line having low-frequency enhanced response characteristics, repetition of the minimum run length which has the shortest bit inversion period makes it difficult for a decoding side to acquire phase lock-up with respect to a
15 received signal. Preferably, repetition of the minimum run length is prevented from occurring as will be mentioned hereafter.

 According to the encoding table 13 in Fig. 2, recurrence of an output code word of "010101" or "101010" causes repetition of the minimum run length which has the shortest bit inversion period.
20 Recurrence of an output code word of "010101" would appear in the case where an input code word $D(k)$ continues to be "7" (decimal) after a current-table selection number $S(k)$ is "0" or "3". Count is made as to the number of times of recurrence of the input code word $D(k)$ and the current-table selection number $S(k)$ which would
25 cause repetition of the minimum run length. The count is to detect given conditions such that $D(k+1)=7$ and $D(k+2)=7$ after $S(k)=0$ and

D(k)=7. In the case where the given conditions are detected,
D(k+1)=13 is used instead of D(k+1)=7. In the sub encoding table
having an ID number of "0", the input code word D(k+1) of "13"
corresponds to an output code word C(k+1) of "000000" which is
5 accompanied with a next-table selection number of "3". The
adopted next-table selection number originally equal to "3" is
changed to "1" so that the sub encoding table having an ID number
of "1" is accessed in response to the input code word D(k+2). In
the sub encoding table having an ID number of "1", the input code
10 word D(k+2) of "7" corresponds to an output code word C(k+2) of
"000100". This design enables the run length limiting rules to be
satisfied, and also enables a decoding side to reproduce repetition of
an original code word D(k) of "7" (decimal).

For example, regarding (1, 9)RLL, (1, 10)RLL, (1, 11)RLL, or
15 (1, 12)RLL, in the case where D(k+1)=7 and D(k+2)=7 after S(k)=0
and D(k)=7, D(k+1)=13 is used instead of D(k+1)=7. Accordingly,
an output code word C(k+1) of "000000" which is accompanied with
a next-table selection number of "3" is read out from the sub
encoding table having an ID number of "0". In addition, the adopted
20 next-table selection number originally equal to "3" is changed to "1"
so that the sub encoding table having an ID number of "1" is
accessed in response to the input code word D(k+2). Therefore, an
output code word C(k+2) of "000100" is read out from the accessed
sub encoding table. In this way, an output code word succession of
25 "000000" and "000100" is generated. A decoding side is designed
to detect a cord word succession of "000000" and "000100", and to

decode the detected cord word succession into a succession of original code words of "7" (decimal). Thus, the input code words $D(k+1)$ and $D(k+2)$ are reproduced. Regarding (1, 8)RLL, $D(k+2)=10, 11, 12, 12, 14$, or 15 is used instead of $D(k+2)=7$. The
5 decoder side can reproduce the input code word $D(k+2)$.

Recurrence of an output code word of "101010" would appear in the following given conditions. When $S(k)=2$ and $D(k)=12$, an output code word $C(k)$ of "101010" is generated. The output code word $C(k)$ is accompanied with a next-table selection number of "2".
10 Then, an input code word $D(k+1)$ of "12" comes, and an output code word $C(k+1)$ of "101010" is generated. The output code word $C(k+1)$ is accompanied with a next-table selection number of "2". Subsequently, an input code word $D(k+2)$ of "12" comes, and an output code word $C(k+2)$ of "101010" is generated. The given
15 conditions are detected by counting the number of times of repetition of the input code word $D(k)$ and the current-table selection number $S(k)$ which would cause repetition of the minimum run length. In the case where the given conditions are detected, the adopted next-table selection number accompanying
20 the output code word $C(k)$ and being originally equal to "2" is changed to "0" so that the sub encoding table having an ID number of "0" is accessed in response to the input code word $D(k+1)$. In the sub encoding table having an ID number of "0", the input code word $D(k+1)$ of "12" corresponds to an output code word $C(k+1)$ of
25 "000000". This design enables the run length limiting rules to be satisfied, and also enables a decoding side to reproduce repetition of

an original code word of "12".

As shown in Fig. 12, the code-word selection detector 121 includes a maximum run length setting circuit 130, a minimum run repetition monitor 131, and a selection detecting circuit 132. The maximum run length setting circuit 130 is connected with the selection detecting circuit 132. The maximum run length setting circuit 130 generates a signal representative of desired run length limiting rules which can be chosen among (1, 7)RLL, (1, 8)RLL, (1, 9)RLL, (1, 10)RLL, (1, 11)RLL, and (1, 12)RLL by a suitable device such as a system controller (not shown). The maximum run length setting circuit 130 informs the selection detecting circuit 132 of the desired run length limiting rules. The minimum run repetition monitor 131 is connected with the selection detecting circuit 132. The minimum run repetition monitor 131 receives the input code word $D(k)$. The minimum run repetition monitor 131 receives the current-table selection number $S(k)$ from the basic encoder 122. The minimum run repetition monitor 131 detects whether or not the previously-mentioned given conditions occur by counting the number of times of repetition of the input code word $D(k)$ and the current-table selection number $S(k)$ which would cause repetition of the minimum run length. When it is detected that the given conditions occur, the minimum run repetition monitor 131 changes at least one of the input code word $D(k)$ and the current-table selection number $S(k)$ in the way same as the previously-mentioned one. The minimum run repetition monitor 131 informs the selection detecting circuit 132 of the change-resultant input code

word $D(k)$ and the change-resultant current-table selection number $S(k)$. On the other hand, when it is detected that the given conditions do not occur, the minimum run repetition monitor 131 passes the input code word $D(k)$ and the current-table selection
5 number $S(k)$ to the selection detecting circuit 132 without changing them. The selection detecting circuit 132 receives the latest selected output code word $C(k-1)$ from the controller 129. The selection detecting circuit 132 detects whether or not an output code word corresponding to the input code word $D(k)$ is
10 uniquely decided, that is, whether or not selecting one from candidate output code words as a final output code word corresponding to the input code word $D(k)$ is required on the basis of the input code word $D(k)$, the current-table selection number $S(k)$, the latest selected output code word $C(k-1)$, and the desired
15 run length limiting rules. The selection detecting circuit 132 outputs either a code-word selection signal or a code-word non-selection signal to the basic encoder 122.

As shown in Fig. 12, the basic encoder 122 includes an address calculation circuit 135, a delay circuit 136, and a distributor
20 137 in addition to the encoding table 13. The address calculation circuit 135 receives the code-word selection signal or the code-word non-selection signal from the code-word selection detector 121. The address calculation circuit 135 receives the input code word $D(k)$. Furthermore, the address calculation circuit 135
25 receives the current-table selection number $S(k)$. The address calculation circuit 135 is connected with the encoding table 13. In

the case where the code-word selection signal is outputted from the code-word selection detector 121, the address calculation circuit 135 computes and generates two different addresses in response to the input code word $D(k)$ and the current-table selection number $S(k)$. The address calculation circuit 135 outputs the generated addresses to the encoding table 13. Two of the four sub encoding tables within the encoding table 13 are accessed in response to the generated addresses. One of the two accessed sub encoding tables has an ID number equal to the current-table selection number $S(k)$. An output code word $C(k)0$ assigned to the input code word $D(k)$ is read out from the sub encoding table having an ID number equal to the current-table selection number $S(k)$. The read-out output code word $C(k)0$ is defined as the first candidate output code word $C(k)0$. An output code word $C(k)1$ assigned to the current input code word $D(k)$ is read out from the other accessed sub encoding table. The read-out output code word $C(k)1$ is defined as the second candidate output code word $C(k)1$. The encoding table 13 is connected with the delay circuit 136 and the distributor 137. The encoding table 13 outputs the first and second candidate output code words $C(k)0$ and $C(k)1$ to the distributor 137. The distributor 137 transmits the first candidate output code word $C(k)0$ to the path "0", that is, the path memory 124 (see Fig. 5). The distributor 137 transmits the second candidate output code word $C(k)1$ to the path "1", that is, the path memory 125 (see Fig. 5). As previously mentioned, one is selected from the first and second candidate output code words $C(k)0$ and $C(k)1$ as a final output code word $C(k)$. A next-table

selection number $S(k+1)$ accompanying the finally-selected output code word $C(k)$ is fed from the encoding table 13 to the delay circuit 136. The delay circuit 136 defers the next-table selection number $S(k+1)$ by a time interval corresponding to one word, thereby generating the current-table selection number $S(k)$. The delay circuit 136 informs the address calculation circuit 135 and the code-word selection detector 121 of the current-table selection number $S(k)$.

In the case where the code-word non-selection signal is outputted from the code-word selection detector 121, the address calculation circuit 135 computes and generates only one address in response to the input code word $D(k)$ and the current-table selection number $S(k)$. The address calculation circuit 135 outputs the generated address to the encoding table 13. One of the four sub encoding tables within the encoding table 13 is accessed in response to the generated address. The accessed sub encoding tables has an ID number equal to the current-table selection number $S(k)$. An output code word $C(k)0$ assigned to the input code word $D(k)$ is read out from the sub encoding table having an ID number equal to the current-table selection number $S(k)$. The read-out output code word $C(k)0$ is defined as the first candidate output code word $C(k)0$. Also, the read-out output code word $C(k)0$ is defined as the second candidate output code word $C(k)1$. The encoding table 13 outputs the first and second candidate output code words $C(k)0$ and $C(k)1$ to the distributor 137. The distributor 137 transmits the first candidate output code word $C(k)0$ to the path "0", that is, the

path memory 124 (see Fig. 5). The distributor 137 transmits the second candidate output code word $C(k)1$ to the path "1", that is, the path memory 125 (see Fig. 5). A next-table selection number $S(k+1)$ accompanying the output code word $C(k)0$ is fed from the encoding table 13 to the delay circuit 136. The delay circuit 136 defers the next-table selection number $S(k+1)$ by a time corresponding to one word, thereby generating the current-table selection number $S(k)$. The delay circuit 136 informs the address calculation circuit 135 and the code-word selection detector 121 of the current-table selection number $S(k)$.

Second Embodiment

Fig. 13 shows a demodulation apparatus 500 according to a second embodiment of this invention. The demodulation apparatus 500 receives an input bit stream divided into segments representative of input code words. The input bit stream is generated by, for example, the modulation apparatus 1 in Fig. 4. The input bit stream corresponds to, for example, the output signal of the NRZI converter 14 in Fig. 4. The demodulation apparatus 500 can reproduce original code words regardless of whether the run length limiting rules used by a modulation side are (1, 7)RLL, (1, 8)RLL, (1, 9)RLL, (1, 10)RLL, (1, 11)RLL, or (1, 12)RLL.

As shown in Fig. 13, the demodulation apparatus 500 includes an NRZI demodulator 501, a sync detector 502, a serial-to-parallel (S/P) converter 503, a word register 504, a code-word decision-information detector 505, a state calculator 506, an address generator 507, and a decoder 508. The NRZI demodulator 501

receives the input bit stream representing a succession of input code words. The NRZI demodulator 501 is connected with the sync detector 502 and the S/P converter 503. The sync detector 502 is connected with the S/P converter 503. The S/P converter 503 is
5 connected with the word register 504 and the state calculator 506. The word register 504 is connected with the code-word decision-information detector 505, the state calculator 506, and the address generator 507. The code-word decision-information detector 505 is connected with the state calculator 506. The state calculator 506
10 is connected with the address generator 507. The address generator 507 is connected with the decoder 508.

The NRZI demodulator 501 subjects the input bit stream to NRZI demodulation (NRZI conversion). The NRZI demodulator 501 outputs the NRZI-demodulation-resultant signal (the NRZI-
15 demodulation-resultant bit stream) to the sync detector 502 and the S/P converter 503.

The sync detector 502 detects every sync word in the NRZI-demodulation-resultant signal. The sync detector 502 generates a word clock signal in response to the detected sync words. The sync
20 detector 502 feeds the generated word clock signal to the S/P converter 503. The S/P converter 503 subjects the NRZI-demodulation-resultant bit stream to serial-to-parallel conversion in response to the word clock signal, thereby periodically generating a 6-bit parallel-form signal segment handled as an input code word
25 C(k). Thus, the S/P converter 503 changes the NRZI-demodulation-resultant bit stream into a sequence of input code words. The S/P

converter 503 outputs the input code word $C(k)$ to the word register 504 and the state calculator 506. The input code word $C(k)$ is written into the word register 504. The input code word $C(k)$ is temporarily stored in the word register 504 before being
5 outputted therefrom as a delayed input code word $C(k-1)$. Specifically, the word register 504 delays the input code word $C(k)$ by a time interval corresponding to one word. The delayed input code word $C(k-1)$ is fed from the word register 504 to the code-word decision-information detector 505, the state calculator 506,
10 and the address generator 507.

The code-word decision-information detector 505 detects a code-word-related decision information in response to the delayed input code word $C(k-1)$. The code-word decision-information detector 505 informs the state calculator 506 of the detected
15 decision information. The state calculator 506 computes an encoding state $S(k)$ from the input code word $C(k)$, the detected decision-information, and the delayed input code word $C(k-1)$. The computed encoding state $S(k)$ corresponds to the sub encoding table used in generating the input code word $C(k)$. In other words,
20 the computed encoding state $S(k)$ is equal to the next-table selection number $S(k+1)$ accompanying the delayed input code word $C(k-1)$ and used in an encoder side (a modulation side). Thus, the next-table selection number $S(k+1)$ accompanying the delayed input code word $C(k-1)$ is recovered. The state calculator 506
25 informs the address generator 507 of the encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the

00000000 10101010 00000000
delayed input code word $C(k-1)$. The address generator 507
produces an address signal in response to the delayed input code
word $C(k-1)$ and the encoding state $S(k)$. The address generator
507 outputs the produced address signal to the decoder 508. The
5 decoder 508 contains a decoding table having an array of 4-bit
output code words at different addresses. The decoding table is
accessed in response to the address signal. One output code word
 $D(k-1)$ at an address corresponding to the address signal is
selected from the output code words in the decoding table. The
10 decoder 508 feeds the selected output code word $D(k-1)$ to an
external as a reproduced original code word $D(k-1)$.

Specifically, the decoding table includes an array of cells each
having a set of an input code word $C(k-1)$, an output code word
 $D(k-1)$, and an encoding state $S(k)$. As previously indicated, the
15 encoding state $S(k)$ corresponds to a next-table selection number
 $S(k+1)$ accompanying the input code word $C(k-1)$. An output code
word $D(k-1)$ can be decided in response to a set of an input code
word $C(k-1)$ and an encoding state $S(k)$ by referring to the
decoding table. An example of the contents of the decoding table is
20 shown in Fig. 14.

Input code words can be grouped into three cases "0", "1",
and "2" according to LSB-side zero run length. The cases "0", "1",
and "2" are given to decision information of "0", "1", and "2",
respectively. Specifically, input code words each having an LSB-side
25 zero run length of "0" are assigned to the case "0", that is, decision
information of "0". Input code words each having an LSB-side zero

run length of "1", "2", or "3" are assigned to the case "1", that is, decision information of "1". Input code words having LSB-side zero run lengths of "4", "5", or "6" are assigned to the case "2", that is, decision information of "2". Each of the input code words in the

5 case "0" (corresponding to decision information of "0") is always followed by an input code word which results from an encoding procedure using the sub encoding table denoted by an ID number of "0" or "1". Each of the input code words in the case "1" (corresponding to decision information of "1") is always followed by

10 an input code word which results from an encoding procedure using the sub encoding table denoted by an ID number of "1", "2", or "3". Each of the input code words in the case "2" (corresponding to decision information of "2") is always followed by an input code word which results from an encoding procedure using the sub encoding

15 table denoted by an ID number of "2" or "3".

The code-word decision-information detector 505 contains a table representative of the previously-mentioned assignment of the input code words to the cases "0", "1", and "2" (decision information of "0", "1", and "2") which depends on LSB-side zero run length.

20 The code-word decision-information detector 505 detects the LSB-side zero run length of the delayed input code word $C(k-1)$. The code-word decision-information detector 505 accesses the assignment table in response to the detected zero run length, and thereby detects the decision information to which the delayed input

25 code word $C(k-1)$ is assigned. The code-word decision-information detector 505 informs the state calculator 506 of the detected

decision information. The state calculator 506 computes an encoding state $S(k)$ from the input code word $C(k)$, the delayed input code word $C(k-1)$, and the detected decision information according to a predetermined algorithm. The computed encoding state $S(k)$ corresponds to the sub encoding table used in generating the input code word $C(k)$. In other words, the computed encoding state $S(k)$ is equal to the next-table selection number $S(k+1)$ accompanying the delayed input code word $C(k-1)$ and used in an encoder side. The state calculator 506 notifies the encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the delayed input code word $C(k-1)$, to the address generator 507. The address generator 507 produces an address signal in response to the delayed input code word $C(k-1)$ and the encoding state $S(k)$. The address generator 507 outputs the produced address signal to the decoder 508. The decoder 508 accesses the decoding table in response to the address signal. An output code word $D(k-1)$ corresponding to the address signal, that is, an output code word $D(k-1)$ corresponding to a set of the delayed input code word $C(k-1)$ and the encoding state $S(k)$, is read out from the decoding table. The decoder 508 feeds the read-out output code word $D(k-1)$ to an external as a reproduced original code word $D(k-1)$.

Fig. 15 shows a succession of input code words of "010000", "001001", "000001", "000101", and "010001". In the case where the input code word $C(k-1)$ of interest is "010000" and the immediately-following input code word $C(k)$ is "001001", since the LSB-side zero run length of the input code word $C(k-1)$ is "4", the

decision information corresponding to the input code word $C(k-1)$ is found to be "2" by referring to the previously-mentioned assignment table. The encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the input code word

5 $C(k-1)$, is found to be "3" according to the predetermined algorithm using the input code word $C(k)$ and the decision information of "2". The input code word $C(k-1)$ of interest is decoded into an output code word $D(k-1)$ of "15" in decimal by referring to the decoding table (see Fig. 14).

10 In the case where the input code word $C(k-1)$ of interest is "001001" and the immediately-following input code word $C(k)$ is "000001", since the LSB-side zero run length of the input code word $C(k-1)$ is "0", the decision information corresponding to the input code word $C(k-1)$ is found to be "0" by referring to the
15 previously-mentioned assignment table. The encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the input code word $C(k-1)$, is found to be "0" according to the predetermined algorithm using the input code word $C(k)$ and the decision information of "0". The input code word $C(k-1)$ of interest
20 is decoded into an output code word $D(k-1)$ of "0" in decimal by referring to the decoding table (see Fig. 14).

In the case where the input code word $C(k-1)$ of interest is "000001" and the immediately-following input code word $C(k)$ is "000101", since the LSB-side zero run length of the input code
25 word $C(k-1)$ is "0", the decision information corresponding to the input code word $C(k-1)$ is found to be "0" by referring to the

previously-mentioned assignment table. The encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the input code word $C(k-1)$, is found to be "1" according to the predetermined algorithm using the input code word $C(k)$ and the
5 decision information of "0". The input code word $C(k-1)$ of interest is decoded into an output code word $D(k-1)$ of "1" in decimal by referring to the decoding table (see Fig. 14).

In the case where the input code word $C(k-1)$ of interest is "000101" and the immediately-following input code word $C(k)$ is
10 "010001", since the LSB-side zero run length of the input code word $C(k-1)$ is "0", the decision information corresponding to the input code word $C(k-1)$ is found to be "0" by referring to the previously-mentioned assignment table. The encoding state $S(k)$, that is, the next-table selection number $S(k+1)$ accompanying the
15 input code word $C(k-1)$, is found to be "0" according to the predetermined algorithm using the input code word $C(k)$ and the decision information of "0". The input code word $C(k-1)$ of interest is decoded into an output code word $D(k-1)$ of "2" in decimal by referring to the decoding table (see Fig. 14).

20 An example of the predetermined algorithm used by the state calculator 506 is as follows.

Algorithm in C-language-based Version:

```
if (decision information == 0 [  
    if (C(k) is in sub encoding table having ID = 0)  
25         S(k)=0;  
    elseif (C(k) is in sub encoding table having ID = 1)
```

S(k)=1;]
if (decision information == 1 [
 if (C(k) is in sub encoding table having ID = 1)
 S(k)=1;
5 elseif (C(k) is in sub encoding table having ID = 2)
 S(k)=2;
 elseif (C(k) is in sub encoding table having ID = 3 || 1)
 S(k)=3;
 elseif (C(k)==0 && C(k-1)==32)
10 S(k)=3;
 elseif (C(k)==0 && C(k-1)==42)
 S(k)=2;]
if (decision information == 2 [
 if (C(k) is in sub encoding table having ID = 3 || 9 || 5 || 2)
15 S(k)=3;
 elseif (C(k) is in sub encoding table having ID = 2 || 10 || 8)
 S(k)=2;
 elseif (C(k)==21)
 S(k)=0;]
20 In the above algorithm: "==" denotes "equal to"; "&&" denotes "and";
 and "||" denotes "or".